TASE USER GUIDE

Author – Satyanand Nalam, University of Virginia

QUICK START

This section provides instructions to generate device and SRAM related data for a particular technology with the default settings.

1. Setup the Cadence environment to enable the Spectre and Ocean simulators.
2. Ensure PERL and Matlab are installed and accessible.
3. cd to the BIN/ directory.
4. Run "perl run.pl –i <path to .ini file>" (e.g. ../../template/ibm45.ini)
5. Output file (output <username>.pdf) – is created in the BIN/ directory.

The default output of TASE consists of the data listed in Table 1.

	Template name	**Description**
Device Related		
1	IDN_vs_IDP	N and P on-current ratio
2	IDVD_N	IV curve
3	IDVD_P	
4	IDVG_N	I_D – V_{GS} curve
5	IDVG_P	
6	Ioff_vs_l_N	I_{OFF} vs. L
7	Ioff_vs_l_P	
8	Ioff_vs_VBS_N	I_{OFF} vs. V_{BS}
9	Ioff_vs_VBS_P	
10	Ioff_vs_VDS_N	I_{OFF} vs. V_{DS}
11	Ioff_vs_VDS_P	
12	Ioff_vs_w_N	I_{OFF} vs. W
13	Ioff_vs_w_P	
14	Ion_vs_l_N	I_{ON} vs. L
15	Ion_vs_l_P	
16	Ion_vs_VBS_N	I_{ON} vs. V_{BS}
17	Ion_vs_VBS_P	
18	Ion_vs_w_N	I_{ON} vs. W
19	Ion_vs_w_P	
20	leakage_N	Leakage currents
21	leakage_P	
SRAM Related		
22	SRAM_DRV	MC sim for DRV

23	SRAM_HSNM	MC sim for Hold SNM
24	SRAM_HSNM_vs_VDD	HSNM vs V_{DD} DC sweep
25	SRAM_Ileak_dist	MC sim for total leakage current
26	SRAM_Ioff_Ileak_vs_lpd	DC sweep of leakage vs. device lengths
27	SRAM_Ioff_Ileak_vs_lpg	
28	SRAM_Ioff_Ileak_vs_lpu	
29	SRAM_Ion_Ioff_vs_VDD	DC sweep of I_{READ} and leakage vs. V_{DD}
30	SRAM_Iread_dist	I_{READ} distribution
31	SRAM_Iread_vs_VWL	DC sweep of I_{READ} vs. V_{WL}
32	SRAM_PUratio	DC sweep of pull-up ratio vs. node '1' voltage
33	SRAM_Qc_alpha	Q_{CRIT} for alpha and neutron strike
34	SRAM_Qc_neutron	
35	SRAM_RSNM	MC sim for Read SNM
36	SRAM_RSNM_vs_VDD	DC sweep of RSNM vs. V_{DD}
37	SRAM_Vbump	DC sweep of cell-ratio vs. node '0' voltage
38	SRAM_Tcrit_mc	MC sim for dynamic write margin (T_{CRIT}) distribution
39	SRAM_WM_SNM	MC sim for write margin distribution and DC sweep for write margin vs. V_{DD} for three methods: 1. Butterfly curve 2. Bitline sweep 3. WL sweep
40	SRAM_WM_SNM_vs_VDD	
41	SRAM_WM_VBL	
42	SRAM_WM_VBL_vs_VDD	
43	SRAM_WM_WL	
44	SRAM_WM_WL_vs_VDD	
45	SRAM_Pfailure_HSNM	Failure probability vs. V_{DD}, V_{MIN} vs. Std. normal quartiles, and V_{MIN} vs. Memory size (for a particular yield) – for hold, write, and read)
46	SRAM_Pfailure_RSNM	
47	SRAM_Pfailure_WNM	

Table 1: Simulation Templates in TASE

TOOL STRUCTURE AND FLOW

Structure:

```
-- TASE
     |
     | -- device/
          | -- TESTS/
          | -- BIN/
     | -- template/
```

device/BIN:

The device/ directory contains the back-end PERL scripts in the BIN/ directory and simulation templates in the TESTS/ directory. The PERL scripts are briefly described in Table 2 below.

PERL script	**Purpose**
run.pl	Top-level script Options: 1. -i <path to .ini file> 2. –noimg (don't create plots) 3. –nopdf (don't create final pdf file)
tpl2scs.pl	Converts the template (input.tpl) file to a .scs or .ocn file by plugging in parameter values from the .ini file
scs2out.pl, ocn2out.pl	Call Spectre or Ocean to run the simulation in a particular technology to produce raw simulation data
out2dat.pl	Converts raw simulation data to MATLAB readable format
dat2img.pl	Collates the MATLAB scripts (plotGen.m) in each template directory and generates plots
img2pdf.pl	Collects the generated plots and creates pdf document of results using LaTeX.

Table 2: Perl scripts in TASE

device/TESTS:

The TESTS/ directory has two kinds of simulation templates – Spectre and Ocean.

Spectre templates:

The following files make up a Spectre template

- <test name>.scs
- input.tpl
- data.ini
- plotGen.m or plotGen.tpl

The <test name>.scs file contains the parametrized netlist of the circuit that is being simulated and initial conditions, if any.

Example:

1. leakage_N/**leakage_N.scs** has the following line

```
N1 (VDS VGS VSS BULK) N_TRANSISTOR width=wdef length=ldef
```

The width and length of the transistor are specified by the wdef and ldef parameters

2. leakage_N/**input.tpl** has the following line that defines the wdef and ldef parameters

```
parameters  ldef=<ldef> \
            wdef=<wdef> \
```

The <ldef> and <wdef> tags are replaced by values for a particular technology, taken from the corresponding .ini file

3. leakage_N/**data.ini** has the following lines which indicate the signals of interest that need to be extracted from the raw Spectre data

```
%dc.dc

dc
N1.NX:d
N1.NX:g
N1.NX:s
VBULK:p
```

4. leakage_N/**plotGen.tpl** plots the data. The tags (e.g. W=<wdef>) are again replaced using values from the .ini file to generate a plotGen.m file for a particular technology.

Ocean templates:
The following files make up an Ocean template

- netlist, netlistHeader, netlistFooter
- input.tpl
- ocnLoader
- plotGen.m/tpl

The netlist* files contain the circuit description. The input.tpl is the .ocn file in a template format and contains the netlist parameter definitions, similar to the spectre templates. The ocnLoader file contains a one-line shell command to load the customized input.ocn file.

templates/:
The template directory contains the following. To add a new process technology the same files described below need to be created:

1. include files for each technology
2. n and p-mos subcircuit definitions for each technology
3. collection of .ini files for various technologies for easy reuse.

Example:
ibm45/ directory contains the following files.

include.scs:

```
include
"/mnt/pdk45/cmos45_11lp/ams2.3_alpha2/amsmodels/spectre/cmos45_11lp.rev
0c.scs" section=base
include
"/mnt/pdk45/cmos45_11lp/ams2.3_alpha2/amsmodels/spectre/cmos45_11lp.rev
0c.scs" section=pre_layout
```

subN.scs:

```
subckt N_TRANSISTOR D G S B

parameters width=70n length=40n pvta=0
```

```
NX (D G S B) nfet w=width l=length nf=1 m=1 \
  as=(120n)*width ad=(120n)*width ps=2*((1)*(120n)+width) \
  pd=2*((1)*(120n)+width) nrd=(70n)/width nrs=(70n)/width \
  sa=(120n) sb=(120n) sd=2*(70n) ptwell=0 pccrit=0 p_vta=pvta
ends N_TRANSISTOR
```

subP.scs:

```
subckt P_TRANSISTOR D G S B

parameters width=70n length=40n pvta=0

PX (D G S B) pfet w=width l=length nf=1 m=1 \
  as=(120n)*width ad=(120n)*width ps=2*((1)*(120n)+width) \
  pd=2*((1)*(120n)+width) nrd=(70n)/width nrs=(70n)/width \
  sa=(120n) sb=(120n) sd=2*(70n) ptwell=0 pccrit=0 p_vta=pvta
ends P_TRANSISTOR
```

The .ini files in the templates/ directory are technology specific and the user provides the values for the parameter tags used in the simulation templates in these files. The parameters defined in the .ini files are described in Table 3 below. In addition, the user also picks a subset of the templates listed in Table 1 to run.

Parameter	**Description**	**Example Value**
MODEL AND DEVICE SPECIFIC		
<minl>	Minimum length	40n
<minw>	Minimum width	70n
<ldef>	Default drawn length	40n
<leff>	Default effective length	40n
<wdef>	Default width	70n
<pdk>	PDK name - should not have underscores	ibm45
<pvdd>	Supply voltage	1
<pvdda>	bitcell supply voltage	1
<pvddwl>	Wordline voltage	1
<pvbp>	Nwell voltage	1
<include>	Path to model files and transistor subckt definitions, relative to test	../../../../template/ibm45/include_mm.scs
<subN>		../../../../template/ibm45/subN.scs
<subP>		../../../../template/ibm45/subP.scs
<nModelName>	nfet model name	nfet
<pModelName>	pfet model name	pfet
VMIN PREDICTION		
<VDDstep>	VDD step for Vmin prediction sims	0.1
<yield>	target yield	0.999
<mem_size>	memory size	1.00E+07
MONTE CARLO		

<mcStartNum>	Iteration starting number	1
<mcrunNum>	Number of iterations	5
<mcType>	Variation type	all
SRAM CELL		
<wpu>	Pull-up width	7.00E-08
<lpu>	Pull-up length	5.50E-08
<wpd>	Pull-down width	2.20E-07
<lpd>	Pull-down length	5.50E-08
<wpg>	Pass-gate width	1.60E-07
<lpg>	Pass-gate length	5.70E-08
WRITE MARGIN TESTS		
<pdat>	Data stored in cell - 0(1) for cell storing 0(1); 2 for evaluating both cases	2
SIMULATOR (OPTIONAL)		
<gmin>	G-min for simulator	1.00E-22
<temp>	Temperature	27

Table 3: Description of parameters defined in .ini file.

Flow:

The tool flow is as follows.

1. Tests to run are parsed from .ini file for a given technology.
2. The parameter tags in input.tpl are replaced by the values from the .ini file.
3. The templates are copied over to device/BIN/<userid>, Spectre/ocean is called and raw data generated for all chosen tests in this work directory.
4. "plotGen" MATLAB script generates plots and/or additional processed data for all tests. The plots for each test are generated in an IMG directory within each test directory.
5. Pdf of results generated using images collected from the IMG directories.

TUTORIAL FOR ADDING A NEW TEST/TEMPLATE

This tutorial describes how the user can expand the existing library of templates.

Spectre tests:

Suppose we want to add a new Spectre template to get the nominal read current of an 8T bitcell. We call the new template 8TSRAM_Iread.

1. Create the corresponding directory in device/TESTS

% mkdir 8TSRAM_Iread

2. Create the following files

a. 8TSRAM_Iread.scs – the netlist containing the 8T cell, initial conditions for the stored data, and the values for various voltage sources, all in parametrized form.

8TSRAM_Iread.scs (parameters – wpu, lpu, pvdd etc.):

```
subckt bitcell_8T BL BLB RBL WWL RWL VDD VSS
      P1 (Q QB VDD VDD) P_TRANSISTOR width=wpu length=lpu
      ...
      ...
ends bitcell_8T
VVDD (VDD 0) vsource dc=pvdd
...
```

b. input.tpl – the spectre input file template containing parameter definitions, analysis and measure statements.

input.tpl:

```
simulator lang=spectre
global 0

parameters lpu=<lpu> wpu=<wpu> pvdd=<pvdd> // netlist parameters
defined
include "<include>" // model files included
include "<subN>" // N_TRANSISTOR defined
include "<subP>" // P_TRANSISTOR defined
include "8TSRAM_Iread.scs" // Include the netlist file

tran ... //transient analysis

save... //save statements
...
```

c. data.ini – the file that lists the signals that need to be extracted from the raw output file

data.ini:

```
%tran.tran // Name of the raw spectre data file from which the selected
signals below are extracted

i(....) // the read current
```

d. plotGen.m – the file that plots the collected data (read current)

Please refer to existing Spectre templates for more detailed examples.

Ocean tests:

Suppose we want to add a test to give the N-curve Readability and Writability metrics for a 6T SRAM. We call this template SRAM_Ncurve.

1. Create the corresponding directory in device/TESTS
% mkdir SRAM_Ncurve

2. Create the following files
a. netlist*

netlist:

```
...
```

```
// Half cell
subckt HALFCELL (IN OUT BL WL VDD VBP VSS VBN)
  MP (OUT IN VDD VBP) P_TRANSISTOR width=wpu length=lpu
  MN (OUT IN VSS VBN) N_TRANSISTOR width=wpd length=lpd
  MT (BL WL OUT VBN) N_TRANSISTOR width=wpg length=lpg
ends HALFCELL

// Devices
ICellAh (QB Q BL WL VDD VBP VSS VBN) HALFCELL
ICellBh (Q QB BLB WL VDD VBP VSS VBN) HALFCELL
...
```

b. input.tpl – the ocean input file template containing parameter definitions, analysis and measure statements.

input.tpl:

```
modelFile(
    '("<include>") ; model files defined
    '("<subN>")
    '("<subP>")
)

...
    desVar(   "wpu" <wpu>     ) % parameters defined
    desVar(   "lpu" <lpu>     )
    desVar(   "wpd" <wpd>     )
    desVar(   "lpd" <lpd>     )
    desVar(   "wpg" <wpg>     )
    desVar(   "lpg" <lpg>     )

    desVar(   "ldef" <ldef> )
    desVar(   "wdef" <wdef> )
    desVar(   "pvdd" VDD_sweep)
    desVar(   "pvbp" <pvbp> )
    desVar(   "pvin" 0      )
    desVar(   "minl" <minl> )
    desVar(   "minw" <minw> )
...

run()
selectResults('dc)

VINA = cross(i("IIN:in") 0 1 "either") % measurements to get N-curve
metrics
VINB = cross(i("IIN:in") 0 2 "either")
VINC = cross(i("IIN:in") 0 3 "either")
...
```

c. ocnLoader – load the customized input.ocn file

ocnloader:

```
load "./input.ocn";
exit
```

d. plotGen.m – the file that plots the collected data (Trip voltages and currents for N-curve read and write metrics).

TROUBLESHOOTING

The following scenarios can cause the tool to hang or stop execution with an error message. Check the following if the tool cannot run.

1. Environment setup

Cadence tools (Spectre, Ocean), PERL, and MATLAB should be in your UNIX path.

2. Model file path

The include files in <technology>/include.scs should be accessible/readable. Also, the <include>, <subN> and <subP> tags should point to the correct path for the include.scs and subN, subP.scs files.

3. Simulation

The run can fail if a simulation fails and does not produce the raw data that is required by the plotGen file to produce the plots. To debug any failed sims, go to device/BIN/<user-id>/<test-name> and run Spectre or Ocean using the generated input.scs or input.ocn file.

One likely cause of failure is when a parameter tag that is defined in the input.tpl file is not specified in the .ini file. This causes the input.scs/ocn file to have a "<...>" string and leads to a syntax error.

4. MATLAB

Any missing packages can cause functions that need them to be undefined. This will cause the plotGen file to error out and the tool to hang or stop running.

Also, if the plotGen file has any parameter tags, they need to be defined in the .ini file, or it will lead to a syntax error.

To debug Matlab errors, edit the collated plotGen.m file that is generated in device/BIN/<user-id> to remove the following lines, and manually run it at the Matlab command line to see what is causing the problem.

Lines to remove:
At the beginning of the file:

```
function plotGen ()
try
```

At the end of the file:

```
exit;
catch
fid = fopen('matlab.err','wt');
```

```
fprintf(fid,'Syntax Error: m-file');
fclose(fid);
exit;
end
```

5. LaTeX

The final source of error can be in converting the generated plots into the final PDF. The latex.log file in device/BIN/<user-id> will give information about any errors encountered during LaTeX processing.